```
for (int i=0; i < N; i++)
    a[i] = b[i] * c[i];
```

FIG. 1a

```
$r1 = 0;
loop (N,5);
$r2 = mem[#b][$r1];
$r3 = mem[#c][$r1];
$r4 = $r2 * $r3;
mem[#a][$r1] = $r4;
$r1 = $r1 + 1;
```

FIG. 1b

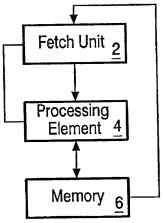

FIG. 2a

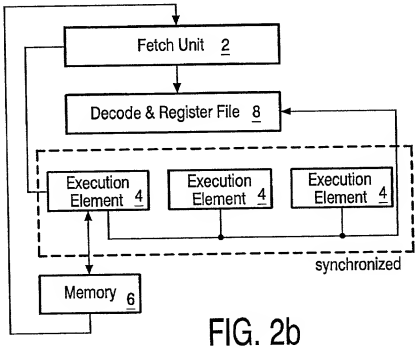

FIG. 2b

```
loop (N,4);          $r1=0,                nop,
$r2 = mem[#b][$r1],  nop,                  nop,
$r3 = mem[#c][$r1],  nop,                  nop,
                     $r4 = $r2 * $r3,      nop,
mem[#a][$r1] = $r4,  $r1 = $r1 + 1,        nop,
```

FIG. 3

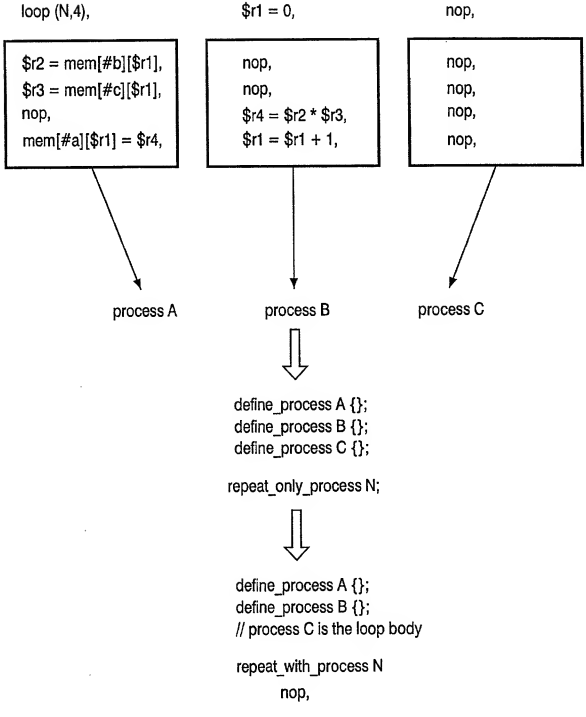
loop (N,4),
$r2 = mem[#b][$r1],
$r3 = mem[#c][$r1],
nop,
mem[#a][$r1] = $r4,
$r1 = 0,
nop,
nop,
$r4 = $r2 * $r3,
$r1 = $r1 + 1,
nop,
nop,
nop,
nop,
nop,
process A
process B
process C
define_process A {};
define_process B {};
define_process C {};
repeat_only_process N;
define_process A {};
define_process B {};
// process C is the loop body
repeat_with_process N
nop,

FIG. 4

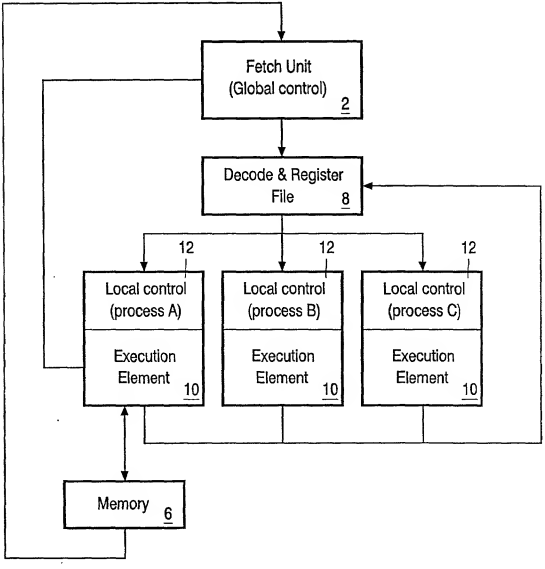
Fetch Unit
(Global control)
2
Decode & Register
File
8
12
12
12
Local control
(process A)
Local control
(process B)
Local control
(process C)
Execution
Element
10
Execution
Element
10
Execution
Element
10
Memory
6

FIG. 5

```
$r2 = mem[#b][$r1];
$r3 = mem[#c][$r1];
nop;
mem[#a][$r1] = $r4;
```

⇒

```
define_process A {
    [$r2,#b, + 1,READ]
    [$r3,#c, + 1,READ]
    [empty_op]
    [$r4,#a, + 1,WRITE]
    }
```

FIG. 6

```
process A {
    $r2 = mem[#b][$r1];
    $r3 = mem[#c][$r1];
    nop;
    mem[#a][$r1] = $r4;
    }
```

⟷ synchronization of every instruction through timing

```
process A {
    nop;
    nop;
    $r4 = $r2 * $r3,
    $r1 = $r1 + 1;
    }
```

FIG. 7a

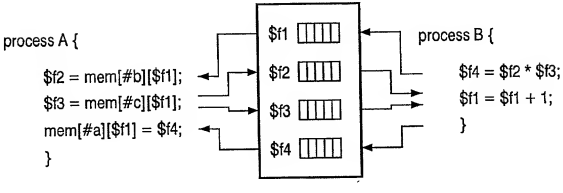

FIG. 7b

```
$r1 = 0;
loop (N,5);
$r2 = mem[#b][$r1];
$r3 = mem[#c][$r1];
$r4 = $r2 * $r3;
mem[#a][$r1] = $r4;
$r1 = $r1 + 1;
```

FIG. 8a

```
// Optional statement for safety
flush_fifo $f2,$f3,$f4;

// This is a single instruction up to a given number of registers involved
// Register $r1 is actually hidden in the local control
define_process A [$f2,Read,#b] [$f3.Read.#c] [$f4,Write,#a]

// This includes process B, process C is not used and the unit left free
repeat_process with B is $f4 = $f2 * $f3;

// Instuction for free units are executed as long as independent from the loop
```

FIG. 8b

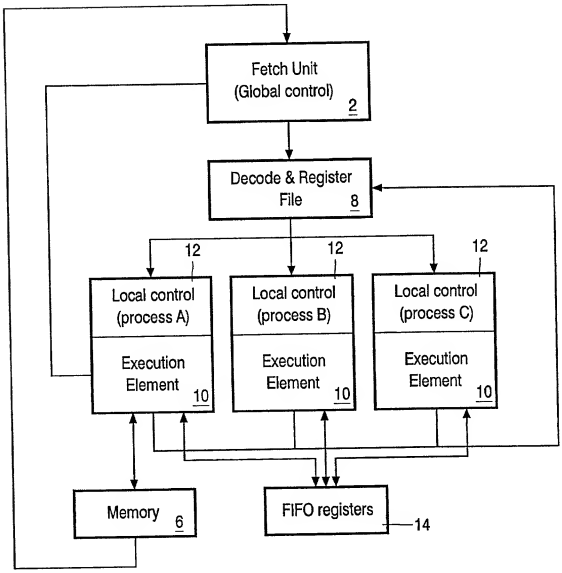
Fetch Unit
(Global control)
2
Decode & Register
File
8
12
12
12
Local control
(process A)
Execution
Element
10
Local control
(process B)
Execution
Element
10
Local control
(process C)
Execution
Element
10
Memory
6
FIFO registers
14

FIG. 9